

**Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
Пермский национальный исследовательский политехнический
университет**

Факультет	Электротехнический
Выпускающая кафедра:	Информационные технологии и автоматизированные системы
Направление подготовки:	09.04.01 «Информатика и вычислительная техника»
Профиль:	Автоматизированные системы обработки информации и управления
Квалификация:	Магистр
Дисциплина:	«Интеллектуальный анализ web-данных»

Отчёт по лабораторной работе №1

На тему: «SOAP-сервисы»

Выполнил: студент группы АСУ4-22-1м

Попов К.М. (_____)

подпись

Проверил:

Ярулин Д. В., к.т.н. (_____)

подпись

Пермь, 2024

Задание

Требуется реализовать программу, которая будет принимать запрос пользователя в относительно свободной форме (например, «столица России»; допустимые категории запросов нужно вывести в начале работы с программой), формировать корректный запрос и затем получать данные при помощи методов любого SOAP-сервиса в данном списке. На выходе необходимо представить понятный пользователю ответ на его запрос.

Программа должна полностью утилизировать весь функционал выбранного SOAP-сервиса.

Теория

SOAP — это протокол, по которому веб-сервисы взаимодействуют друг с другом или с клиентами. Название происходит от сокращения Simple Object Access Protocol («простой протокол доступа к объектам»). SOAP API — это веб-сервис, использующий протокол SOAP для обмена сообщениями между серверами и клиентами. При этом сообщения должны быть написаны на языке [XML](#) в соответствии со строгими стандартами, иначе сервер вернет ошибку.

Протокол SOAP был представлен в 1998 году и быстро стал одним из главных стандартов веб-служб, когда Microsoft продвигала платформу .NET, приложения которой взаимодействовали с помощью SOAP API. Сейчас протокол и API уступают по популярности архитектурному стилю REST. Но веб-приложения, использующие SOAP API, все еще пользуются спросом, особенно в банковском и телекоммуникационном секторах.

SOAP может использоваться с протоколами SMTP, FTP, [HTTP](#), HTTPS. Чаще всего — с HTTP как с наиболее универсальным: его поддерживают все браузеры и серверы. Корректное SOAP-сообщение состоит из нескольких структурных элементов: Envelope, Header, Body и Fault.

Envelope («конверт»). Это корневой элемент. Определяет [XML-документ](#) как сообщение SOAP с помощью пространства имен `xmlns_soap=»http://www.w3.org/2003/05/soap-envelope/»`. Если в определении будет указан другой адрес, сервер вернет ошибку.

Header («заголовок»). Включает в себя атрибуты сообщения, связанные с конкретным приложением (аутентификация, проведение платежей и так далее). В заголовке могут использоваться три атрибута, которые указывают, как принимающая сторона должна обрабатывать сообщение, — *mustUnderstand*, *actor* и *encodingStyle*. Значение *mustUnderstand* — 1 или 0 — говорит принимающему приложению о том, следует ли распознавать заголовок в обязательном или опциональном порядке. Атрибут *actor* задает конкретную конечную точку для сообщения. Атрибут *encodingStyle* устанавливает

специфическую кодировку для элемента. По умолчанию SOAP-сообщение не имеет определенной кодировки.

Body («тело»). Сообщение, которое передает веб-приложение. Может содержать запрос к серверу или ответ от него. Пример сообщения, которое запрашивает стоимость ноутбука в онлайн-магазине:

```
<?xml version="1.0"?> <soap:Envelope
xmlns_soap="http://www.w3.org/2003/05/soap-envelope/"
soap_encodingStyle="http://www.w3.org/2003/05/soap-encoding"> <soap:Body>
<m:GetPrice xmlns_m="https://online-shop.ru/prices"> <m:Item>Dell Vostro 3515-
5371</m:Item> </m:GetPrice> </soap:Body> </soap:Envelope>
```

Пример ответа сервера онлайн-магазина:

```
<?xml version="1.0"?> <soap:Envelope
xmlns_soap="http://www.w3.org/2003/05/soap-envelope/"
soap_encodingStyle="http://www.w3.org/2003/05/soap-encoding"> <soap:Body>
<m:GetPriceResponse xmlns_m="https://online-shop.ru/prices">
<m:Price>37299</m:Price> </m:GetPriceResponse> </soap:Body>
</soap:Envelope>
```

Fault («ошибка»). Опциональный элемент. Передает уведомление об ошибках, если они возникли в ходе обработки сообщения. Может содержать вложенные элементы, которые проясняют причину возникновения ошибки:

- faultcode — код неполадки;
- faultstring — «человекопонятное» описание проблемы;
- faultactor — информация о программном компоненте, который вызвал ошибку;
- detail — дополнительные сведения о месте возникновения неполадки.

SOAP — протокол, а REST — архитектурный стиль, набор правил по написанию кода. REST был представлен в 2000 году. К этому времени недостатки SOAP были очевидны:

- объемные сообщения;

- поддержка только одного формата — XML;
- схема работы по принципу «один запрос — один ответ»;
- смена описания веб-сервиса может нарушить работу клиента.

Разработчик стиля REST Рой Филдинг учел недостатки SOAP. REST поддерживает несколько форматов помимо XML: [JSON](#), TXT, CSV, [HTML](#). Вместо создания громоздкой структуры XML-запросов при использовании REST чаще всего можно передать нужный URL. Эти особенности делают стиль REST простым и понятным, а приложения и веб-сервисы, использующие его, отличаются высокой производительностью и легко масштабируются.

Пример простого URL-запроса, возвращающего результаты поиска по ключевому слову DNA («ДНК»), можно посмотреть в международной базе научных статей.

Несмотря на простоту использования, у REST есть ряд недостатков, которые отсутствуют у SOAP:

- при использовании REST сложнее обеспечить безопасность конфиденциальных данных;
- трудности с проведением операций, которым необходимо сохранение состояния. Как, например, в случае с корзиной в онлайн-магазине, которая должна сохранять добавленные товары до момента оплаты.

В каких случаях используют SOAP:

- Асинхронная обработка и последующий вызов. Стандарт SOAP 1.2 обеспечивает клиенту гарантированный уровень надежности и безопасности.
- Формальное средство коммуникации. Если клиент и сервер имеют соглашение о формате обмена, то SOAP 1.2 предоставляет жесткие спецификации для такого типа взаимодействия. Пример — сайт онлайн-покупок, на котором пользователи добавляют товары в корзину перед оплатой. Предположим, что есть веб-служба, которая выполняет

окончательный платеж. Может быть достигнуто соглашение, что веб-сервис будет принимать только название товара, цену за единицу и количество. Если сценарий существует, лучше использовать протокол SOAP.

- Операции с состоянием. Если приложение требует, чтобы состояние сохранялось от одного запроса к другому, то стандарт SOAP 1.2 предоставляет структуру для поддержки таких требований.

Реализация

Откроем VSCode. Чтобы не навредить системе, создадим виртуальную среду .venv, где будут храниться все файлы и зависимости и активируем её. Установим библиотеку для работы с SOAP-сервисами zeep в python3 (рис. 1).

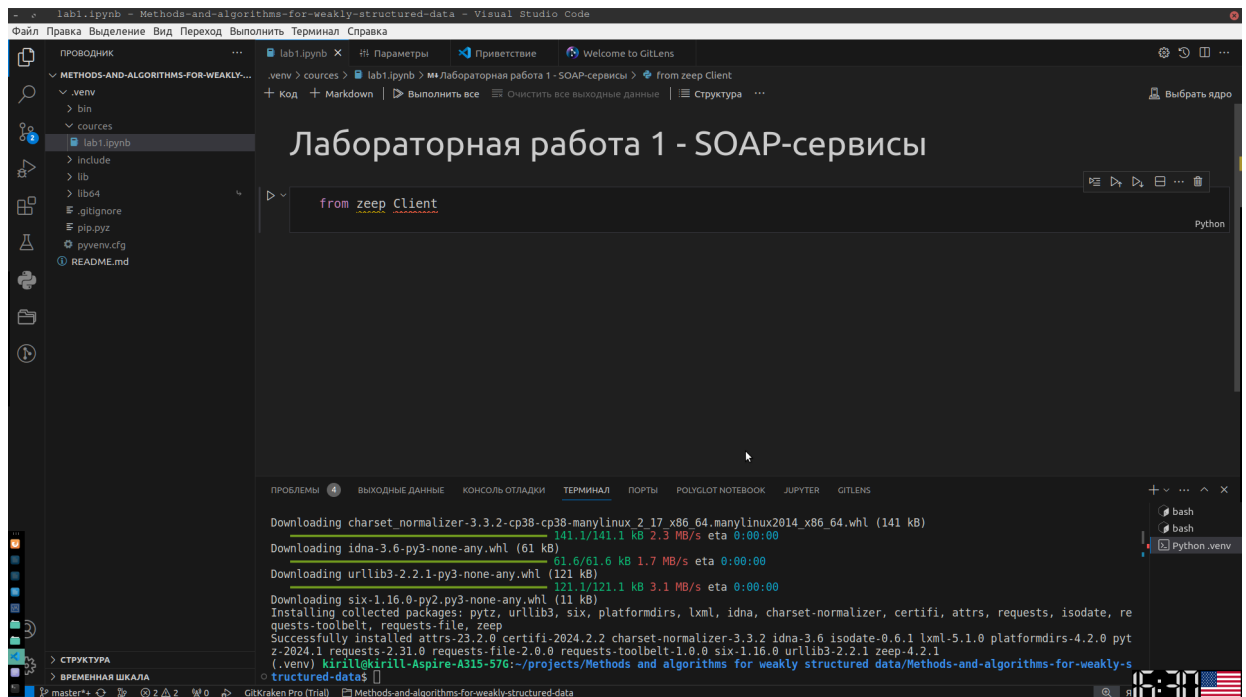


Рисунок 1 - Установка библиотеки zeep

На рис. 2 и 3 представлен исходный код программы.

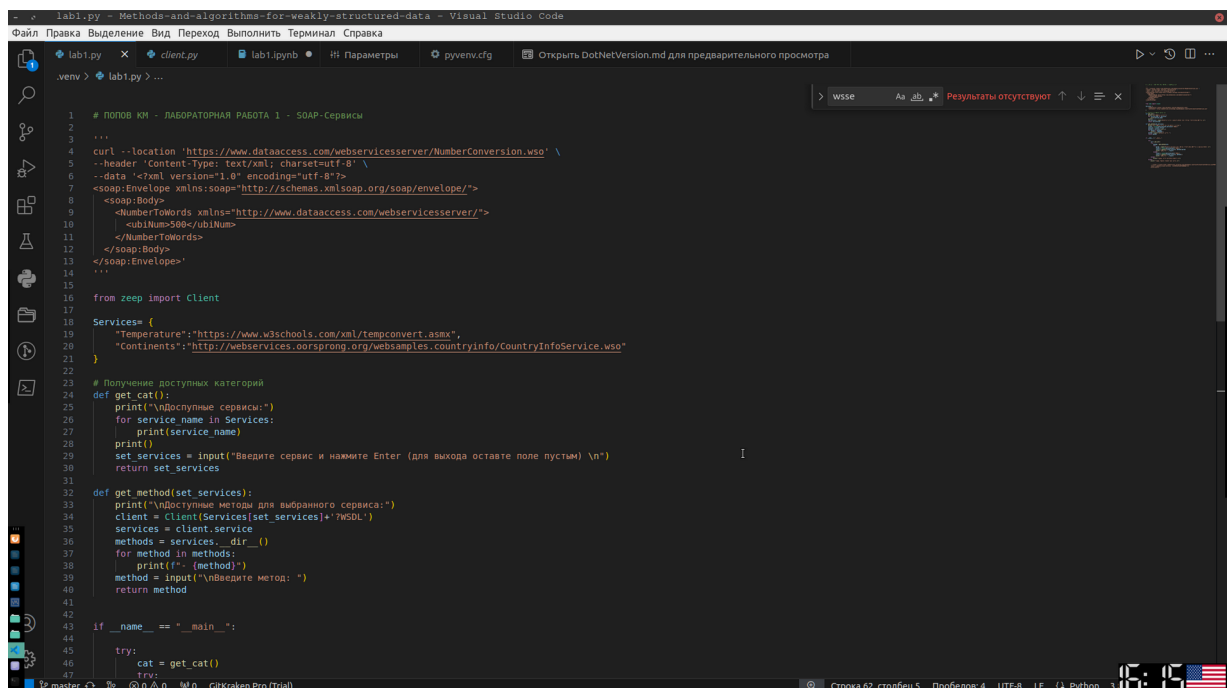
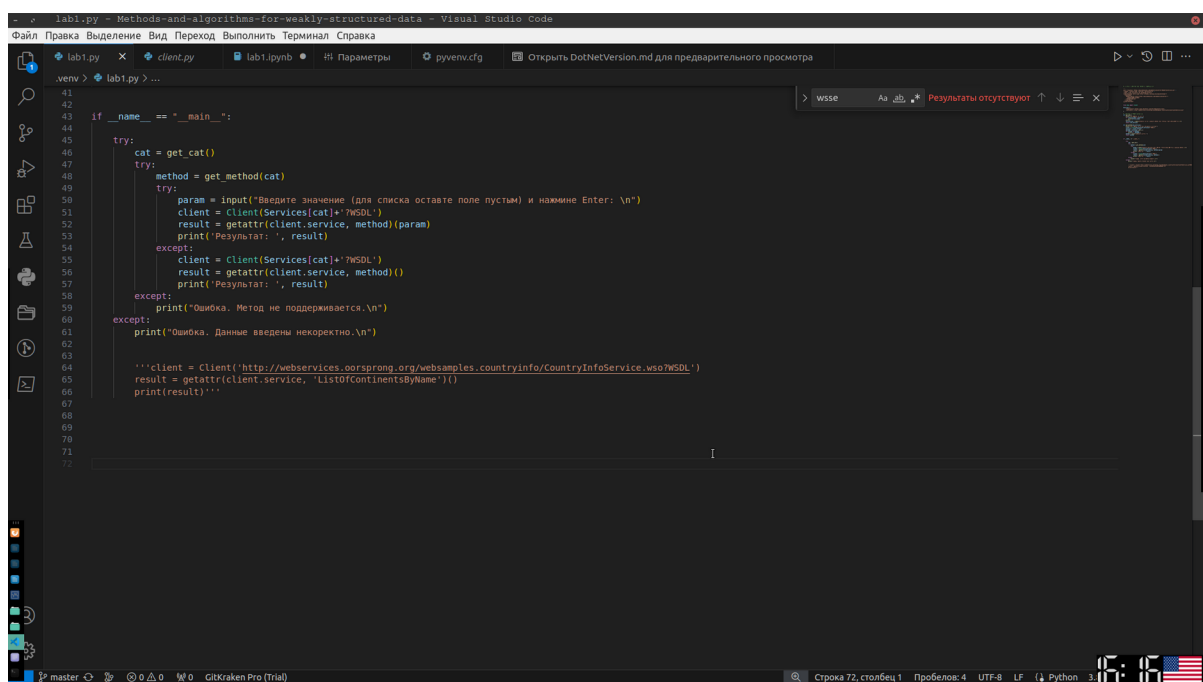


Рисунок 2 - Исходный код, часть 1



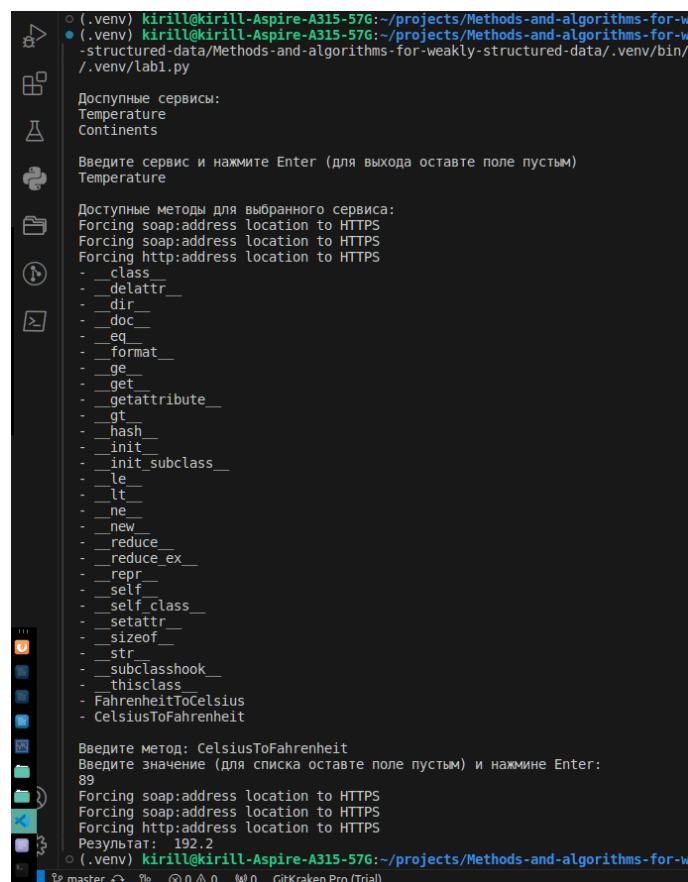
```
lab1.py - Methods-and-algorithms-for-weakly-structured-data - Visual Studio Code
Файл Правка Выделение Вид Переход Выполнить Терминал Справка

.venv > lab1.py > ...

41
42
43
44 if __name__ == "__main__":
45
46     try:
47         cat = get_cat()
48         try:
49             method = get_method(cat)
50             param = input("Введите значение (для списка оставьте поле пустым) и нажмите Enter: \n")
51             client = Client(Services[cat]+'?WSDL')
52             result = getattrib(client.service, method)(param)
53             print("Результат: ", result)
54         except:
55             client = Client(Services[cat]+'?WSDL')
56             result = getattrib(client.service, method)()
57             print("Результат: ", result)
58     except:
59         print("Ошибка. Метод не поддерживается.\n")
60     except:
61         print("Ошибка. Данные введены некорректно.\n")
62
63
64 '''client = Client('http://webservicex.microsoft.com/websamples/countryinfo/CountryInfoService.wsdl')
65 result = getattrib(client.service, 'ListOfContinentsByRegion')()
66 print(result)'''
67
68
69
70
71
72
```

Рисунок 3 - Исходный код, часть 2

Программа использует 2 сервиса «Temperature» и «Continents». На рис. 4 представлены результаты работы сервиса «Temperature».



```
(.venv) kirill@kirill-Aspire-A315-57G: ~/projects/Methods-and-algorithms-for-w
(.venv) kirill@kirill-Aspire-A315-57G: ~/projects/Methods-and-algorithms-for-w
-structured-data/Methods-and-algorithms-for-weakly-structured-data/.venv/bin/
/.venv/lab1.py

Доступные сервисы:
Temperature
Continents

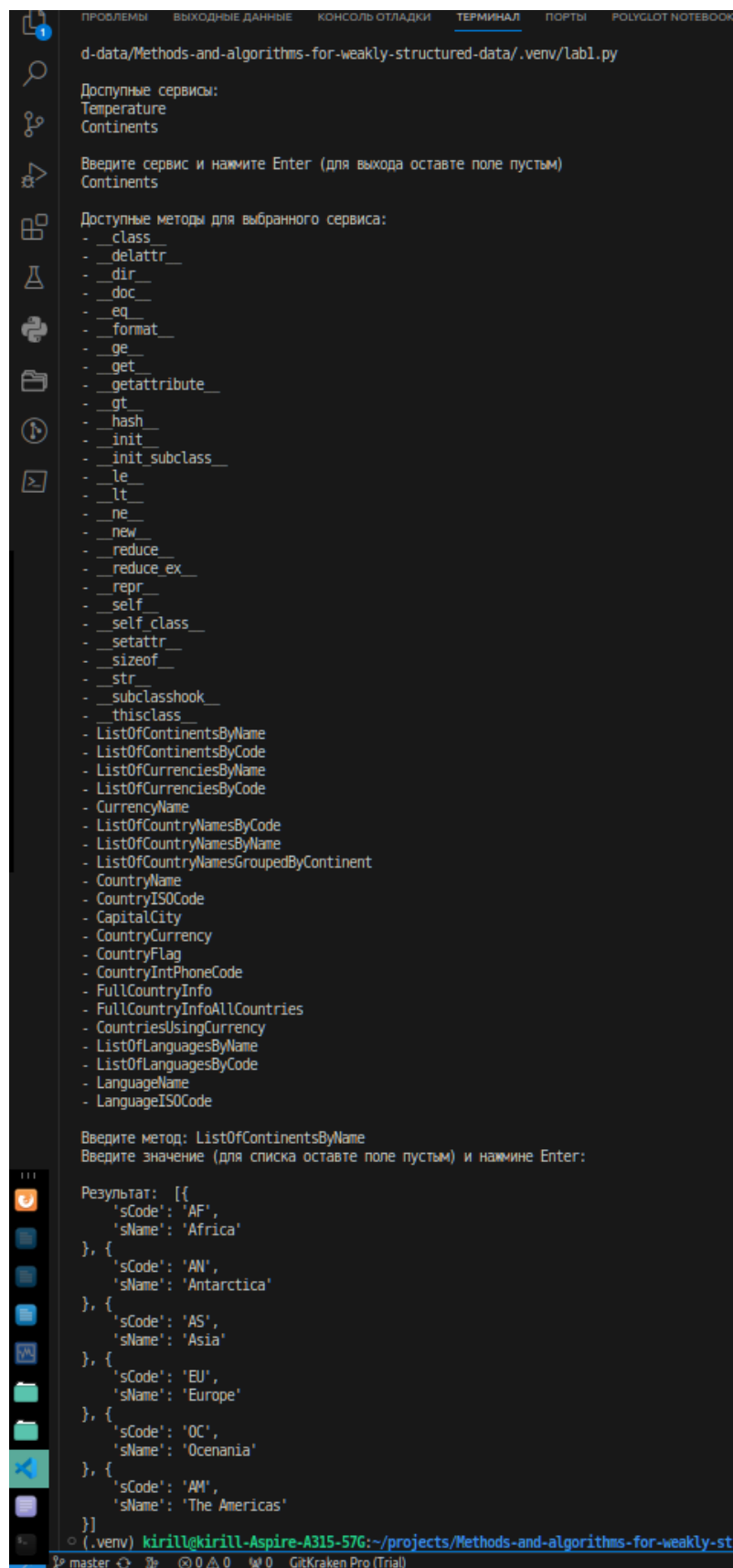
Введите сервис и нажмите Enter (для выхода оставьте поле пустым)
Temperature

Доступные методы для выбранного сервиса:
Forcing soap:address location to HTTPS
Forcing soap:address location to HTTPS
Forcing http:address location to HTTPS
- __class__
- __delattr__
- __dir__
- __doc__
- __eq__
- __format__
- __ge__
- __get__
- __getattr__
- __gt__
- __hash__
- __init__
- __init_subclass__
- __le__
- __lt__
- __ne__
- __new__
- __reduce__
- __reduce_ex__
- __repr__
- __self__
- __self_class__
- __setattr__
- __sizeof__
- __str__
- __subclasshook__
- __thisclass__
- FahrenheitToFahrenheit
- CelsiusToFahrenheit

Введите метод: CelsiusToFahrenheit
Введите значение (для списка оставьте поле пустым) и нажмите Enter:
89
Forcing soap:address location to HTTPS
Forcing soap:address location to HTTPS
Forcing http:address location to HTTPS
Результат: 192.2
(.venv) kirill@kirill-Aspire-A315-57G: ~/projects/Methods-and-algorithms-for-w
```

Рисунок 4 - Сервис «Temperature»

На рис. 5 представлено взаимодействие с сервисом «Continents».



```
d-data/Methods-and-algorithms-for-weakly-structured-data/.venv/lab1.py

Доступные сервисы:
Temperature
Continents

Введите сервис и нажмите Enter (для выхода оставте поле пустым)
Continents

Доступные методы для выбранного сервиса:
- __class__
- __delattr__
- __dir__
- __doc__
- __eq__
- __format__
- __ge__
- __get__
- __getattr__
- __gt__
- __hash__
- __init__
- __init_subclass__
- __le__
- __lt__
- __ne__
- __new__
- __reduce__
- __reduce_ex__
- __repr__
- __self__
- __self_class__
- __setattr__
- __sizeof__
- __str__
- __subclasshook__
- __thisclass__
- ListOfContinentsByName
- ListOfContinentsByCode
- ListOfCurrenciesByName
- ListOfCurrenciesByCode
- CurrencyName
- ListOfCountryNamesByCode
- ListOfCountryNamesByName
- ListOfCountryNamesGroupedByContinent
- CountryName
- CountryISOCode
- CapitalCity
- CountryCurrency
- CountryFlag
- CountryIntPhoneCode
- FullCountryInfo
- FullCountryInfoAllCountries
- CountriesUsingCurrency
- ListOfLanguagesByName
- ListOfLanguagesByCode
- LanguageName
- LanguageISOCode

Введите метод: ListOfContinentsByName
Введите значение (для списка оставте поле пустым) и нажмите Enter:

Результат: [{
  'sCode': 'AF',
  'sName': 'Africa'
}, {
  'sCode': 'AN',
  'sName': 'Antarctica'
}, {
  'sCode': 'AS',
  'sName': 'Asia'
}, {
  'sCode': 'EU',
  'sName': 'Europe'
}, {
  'sCode': 'OC',
  'sName': 'Oceania'
}, {
  'sCode': 'AM',
  'sName': 'The Americas'
}]

(.venv) kirill@kirill-Aspire-A315-57G:~/projects/Methods-and-algorithms-for-weakly-st
master 0 0 0 0 0 GitKraken Pro (Trial)
```

Рисунок 5 - Расаба с «Continents»

Заключение

В результате выполнения лабораторной работы №1 была реализована программа для взаимодействия с SOAP-сервисами. Приведён пример работы с двумя сервисами «Continents» и «Temperature».